

APPENDIX A

% BEGINNING OF PSEUDO CODE

```
5      % compute scale factor A, and time constants a, b from physical system
      % parameters

      A = Vmax * Kt / (Re * Rm + Kt * Kb) * l * k;

10     p1 = 1/Jm/Ie * (-Ie * Rm - Re * Jm + sqrt(Ie^2 * Rm^2 - 2 * Re * Rm * Ie * Jm
      + Re^2 * Jm^2 - 4 * Kt * Kb * Ie * Jm)) / 2;
      p2 = 1/Jm/Ie * (-Ie * Rm - Re * Jm - sqrt(Ie^2 * Rm^2 - 2 * Re * Rm * Ie * Jm
      + Re^2 * Jm^2 - 4 * Kt * Kb * Ie * Jm)) / 2;

15     a = max(-p1,-p2)
      b = min(-p1,-p2)

      % make initial guesses for step durations

20     et1 = 1;
      et2 = .005;
      et3 = 1;

      % set maximum iteration count

25     Nmax = 1000;

      for j = 1:Nmax
          % save old values of step time intervals
          et3old = et3;
```

```

et2old = et2;
et1old = et1;

% iterate for switch times using fixed voltage level Vmax

5
et3 = -log(1.0 / 2.0 - exp(-et1 * a) / 2 + exp(-et2 * a)) / a;
et2 = 1/b * log(2.0) + 3 * et3 - 1/b * log(2 * exp(1/A * b * X) * exp(et3
* b) - sqrt( 4.0) * sqrt(exp(1/A * b * X)) * exp(et3 * b) *
sqrt(exp(1/A * b * X)+exp(et3 * b)^2 - 2 * exp(et3 * b)));
et1 = - (-2 * A * et2 + 2 * A * et3 - X) / A;

10
if norm([et3old - et3 et2old - et2 et1old - et1], inf) <= eps * 2
    break
end

15
if j==Nmax
    error(['error - failure to converge after ', num2str(Nmax), '
iterations'])
end
end

20
% round up pulse duration to nearest sample interval,
% convert to intervals between steps to make sure that voltage
% requirements will not increase (beyond Vmax).

25
dt1=ceil((et1 - et2) / dt) * dt;
dt2=ceil((et2 - et3) / dt) * dt;
dt3=ceil((et3) / dt) * dt;

et123 = [et1, et2, et3]
30
% convert back to total step duration.

```

```

et1 = dt1 + dt2 + dt3;
et2 = dt2 + dt3;
et3 = dt3;

5      % In the following, the original constraints equations involving XF1, XF2,
% and XF3 have been modified to include a variable voltage level applied
at
% each step (instead of the fixed maximum (+-) Vmax).

10     % The original equations for XF1, XF2, and XF3 follow:
% XF1(tend) = V0F1(tend - t0) - 2V0F1(tend - t1) + 2V0F1(tend - t2)
% XF2(tend) = V0F2(tend - t0) - 2V0F2(tend - t1) + 2V0F1(tend - t2)
% XF3(tend) = V0F3(tend - t0) - 2V0F2(tend - t1) + 2V0F1(tend - t2)

15     % And the modified equation including adjustable relative levels of
voltage
% L1, L2 and L3 are:
% XF1(tend) = L1V0F1(tend - t0) - L2V0F1(tend - t1) + L3V0F1(tend - t2)
% XF2(tend) = L1V0F2(tend - t0) - L2V0F2(tend - t1) + L3V0F1(tend - t2)
20     % XF3(tend) = L1V0F3(tend - t0) - L2V0F2(tend - t1) + L3V0F1(tend - t2)

% And the corresponding constraint equations are:
% XF1(tend) = Finalpos
% XF2(tend) = 0
25     % XF3(tend) = 0

% Where all of the times indicated have discrete values, e.g.
corresponding to
% the controller update rate.

30

```

% It should be noted that after the digital switch times are fixed, the constraint

% equations derived from the equations above form a linear set of equations in

5 % the unknown relative voltage levels L1, L2 and L3 and any standard linear

% method can be used to solve for the relative voltage levels. In the equations

10 % for (L1, L2 and L3) that follow, the solution was obtained by algebraic % means (and are not particularly compact.)

% compute new relative voltage step levels

% L1, L2 and L3 are nominally assigned to "1", "-2" and "+2", respectively

15 s1 = X * (exp(-et3 * b) * exp(-et2 * a) + exp(-et3 * a) + exp(-et2 * b) - exp(-et2 * b) * exp(-et3 * a) - exp(-et2 * a) - exp(-et3 * b));

s2 = 1 / (et2 * exp(-et1 * b) * exp(-et3 * a) + exp(-et2 * b) * et3 * exp(-et1 * a) - et2 * exp(-et3 * a) - et2 * exp(-et1 * b) - et3 * exp(-et1 * a) - exp(-et2 * b) * et3 + exp(-et3 * b) * et1 * exp(-et2 * a))

20 a) + exp(-et3 * a) * et1 + exp(-et2 * b) * et1 - exp(-et2 * b) * et1 * exp(-et3 * a) - et3 * exp(-et1 * b) * exp(-et2 * a) - exp(-et2 * a) * et1 - exp(-et3 * b) * et1 - exp(-et3 * b) * et2 * exp(-et1 * a) + et3 * exp(-et1 * b) + et2 * exp(-et1 * a) + exp(-et3 * b) * et2 + et3 * exp(-et2 * a)) / A;

25

L1 = s1 * s2;

s1 = 1 / (et2 * exp(-et1 * b) * exp(-et3 * a) + exp(-et2 * b) * et3 * exp(-et1 * a) - et2 * exp(-et3 * a) - et2 * exp(-et1 * b) - et3 * exp(-et1 * a) - exp(-et2 * b) * et3 + exp(-et3 * b) * et1 *

30

```

exp(-et2 * a) + exp(-et3 * a) * et1+exp(-et2 * b) * et1 -
exp(-et2 * b) * et1 * exp(-et3 * a) - et3 * exp(-et1 * b) *
exp(-et2 * a) - exp(-et2 * a) * et1 - exp(-et3 * b) * et1 - exp(-et3 *
b) * et2 * exp(-et1 * a) + et3 * exp(-et1 * b) + et2 * exp(-et1 * a) +
exp(-et3 *b ) * et2 + et3 * exp(-et2 * a)) * X;
5      s2 = (exp(-et2 * b) * exp(-et1 * a) - exp(-et1 * a) - exp(-et2 * b) -
exp(-et1 * b) * exp(-et2 * a) + exp(-et1 * b) + exp(-et2 * a)) / A;
L3 = s1*s2;

10     s1 = exp(-et1 * a) - exp(-et3 * a) + exp(-et3 * b) - exp(-et1 * b) -
exp(-et3 * b) * exp(-et1 * a) + exp(-et1 * b) * exp(-et3 * a);
s2 = X / (et2 * exp(-et1 * b) * exp(-et3 * a) + exp(-et2 * b) * et3 *
exp(-et1 * a) - et2 * exp(-et3 * a) - et2 * exp(-et1 * b) - et3 *
exp(-et1 * a) - exp(-et2 * b) * et3 + exp(-et3 * b) * et1 * exp(-et2 *
15    a) + exp(-et3 * a) * et1 + exp(-et2 * b) * et1 - exp(-et2 * b) * et1 * exp(-
et3 * a) - et3 * exp(-et1 * b) * exp(-et2 * a) - exp(-et2 * a) * et1-exp(-et3 *
b) * et1 - exp(-et3 * b) * et2 * exp(-et1 * a) + et3 *
exp(-et1 * b) + et2 * exp(-et1 * a) + exp(-et3 * b) * et2 + et3 *
exp(-et2 * a)) / A;
20      L2 = s1 * s2;

```

% convert accumulated voltage steps to sequential voltage level

```

V1 = Vmax * (L1);
25    V2 = Vmax * (L1 + L2);
V3 = Vmax * (L1 + L2 + L3);

```

% END OF PSEUDO CODE